САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ

Рябов Виктор Михайлович, Бурова Ирина Герасимовна,

Кальницкая Марина Алексеевна, Малевич Александр Владиславович,

Лебедева Анастасия Владимировна, Доронина Александра Геннадьевна

О ЧИСЛЕННОМ РЕШЕНИИ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ С ПЛОХО ОБУСЛОВЛЕННЫМИ МАТРИЦАМИ

Учебно-методическое пособие

Дисциплина [003616] «Вычислительная математика»

по направлению 09.03.04 Программная инженерия

учебный план рег. № 19/5080/1

САНКТ-ПЕТЕРБУРГ

2019

Рябов Виктор Михайлович, Бурова Ирина Герасимовна,

Кальницкая Марина Алексеевна, Малевич Александр Владиславович,

Лебедева Анастасия Владимировна, Доронина Александра Геннадьевна

О ЧИСЛЕННОМ РЕШЕНИИ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ С ПЛОХО ОБУСЛОВЛЕННЫМИ МАТРИЦАМИ

Рецензент профессор, д.ф.-м.н. Демьянович Юрий Казимирович

Печатается по рекомендации УМК по УГСН 09.00.00 Информатика и вычислительная техника

от 28 марта 2019 г

Аннотация

Представлены результаты численного решения систем линейных алгебраических уравнений (СЛАУ) с симметричными и несимметричными плохо обусловленными матрицами методом регуляризации. Рассматриваются положительно определенные, а также осцилляционные матрицы. В статье показано, что для регуляризации вычислительного процесса по методу Тихонова достаточно заменить симметричную положительно определенную матрицу A_n системы матрицей $A_n + \alpha E_n$ где E_n – единичная матрица, а α – некоторое положительное число (параметр регуляризации), которое стремится к нулю.

Ключевые слова: плохо обусловленные системы линейных алгебраических уравнений, матрицы Гильберта, параметр регуляризации

1. ВВЕДЕНИЕ

При решении различных задач возникает необходимость решать плохо обусловленные системы линейных алгебраических уравнений (СЛАУ) с положительно определенными симметричными матрицами. Такие системы возникают, например, в теории и практике приближения функций [1], при решении задач математической физики вариационными методами [2]. При этом «качество» СЛАУ определяется разумным выбором координатных систем [2]. Приведем пример классической задачи подобного типа: пусть функция $f(x)$ аппроксимируется алгебраическим многочленом в метрике пространства $L_2(0,1)$: если мы возьмем полином в виде $\sum_{i=0}^{n} c_i x^i$ и определим погрешность аппроксимации как $E = \int_0^1 \left(\sum_{i=0}^{n} c_i x^i - f(x)\right)^2 dx$, то из условия минимизации E придем к СЛАУ с матрицей Гильберта $H_n = (1/(i+j-1))_{i,j=1}^{n}$. Аналогичные системы линейных алгебраических уравнений возникают при решении обыкновенных дифференциальных уравнений методом Ритца. Эти матрицы размерности n являются симметричными и положительно определенными, но при неограниченном увеличении n наименьшее собственное значение может стремиться к нулю, что приводит к неустойчивости решения.

Обычно для получения надежного решения используют методы регуляризации. Общей стратегией является использование стабилизатора Тихонова [3] или его модификаций [4-9], либо представление искомого решения в виде ортогональной суммы двух векторов, один из которых определяется устойчиво, а для поиска второго необходима некая процедура стабилизации [6]. В настоящей статье рассматриваются особенности численного решения СЛАУ с положительно определенной симметричной матрицей с использованием регуляризации. В следующих разделах показано, что для регуляризации вычислительного процесса по методу Тихонова достаточно заменить матрицу системы A_n матрицей $A_n + \alpha E_n$, где E_n –

единичная матрица, а α – положительное число (параметр регуляризации), стремящееся к нулю и согласованное с точностью исходных данных. Таким образом, мы уменьшаем число обусловленности системы линейных алгебраических уравнений, что увеличивает устойчивость.

2. ПОСТАНОВКА ЗАДАЧИ

Пусть A – невырожденная вещественная квадратная матрица порядка n. В этом случае решение СЛАУ $Az = f$ существует и единственно. Известны различные модификации метода Гаусса для решения СЛАУ, например, метод Гаусса с выбором ведущего элемента (в столбце или в матрице) и другие. Предположим, что число обусловленности $\mathrm{cond}\,(A) = \|A\| \cdot \|A^{-1}\|$ матрицы A велико, т.е. матрица системы уравнений плохо обусловлена. Решение плохо обусловленных СЛАУ методом Гаусса не всегда дает удовлетворительное решение. Например, пусть

$$\mathrm{A} = \begin{pmatrix} 0.0000001 & 333 & 555 \\ 33333333 & 1 & 70 \\ 55555555 & 70 & 32 \end{pmatrix},\ f = \begin{pmatrix} 888 \\ 33333404 \\ 55555657 \end{pmatrix}.$$

Число обусловленности матрицы A приближенно равно 10^7. Решая эту систему методом Гаусса без перестановок (с помощью программы, написанной на C++ с вещественными числами типа double), получим $z = (1.0,\ \ 1.555556,\ \ 0.666667)^T$, что существенно отличается от точного решения $(1.0,\ \ 1.0,\ \ 1.0)^T$. Подобные примеры были рассмотрены в [8]. Эти примеры показывают, что в процессе решения необходимо избегать деления на малые по абсолютной величине элементы. Избежать эту ситуацию помогает использование модифицированного метода Гаусса, заключающегося в выборе ведущего элемента, являющегося наибольшим по абсолютному значению элементом в столбце (стратегия Уилкинсона) или по

всей матрице (стратегия полного упорядочения Жордана) [4]. Применение метода Гаусса с выбором ведущего элемента по столбцам дает решение $z = (1.0,\ 1.0,\ 1.0)^T$. Если система уравнений является плохо обусловленной (например, в случае СЛАУ с матрицей Гильберта H_n), то практически невозможно получить приемлемое решение СЛАУ с помощью известных методов (прямыми методами, такими как метод Гаусса, метод квадратного корня, итерационными методами и др.).

В таблице 1 приведены числа обусловленности матриц Гильберта порядков от 3 до 20, полученные с помощью пакета Maple (Digits=50). В таблице 2 представлены решения СЛАУ $Az = f$ с матрицами Гильберта H_n, полученные методом Гаусса (стратегия Уилкинсона). Здесь представлены решения, полученные при решении систем уравнений для $n = 10, 12, 14, 20$, вычисленные в C++ с числами типа double. Решения, представленные в таблице 2, далеки от истинных решений. В следующих разделах представлен метод регуляризации, с помощью которого получены решения исходной системы $Az = f$ с матрицами Гильберта $A = H_n$ при использовании нормы $\| H_n \| = \max_i \sum_{j=1}^{n} | h_{ij} |$.

Таблица 1 – Числа обусловленности матриц Гильберта H_n.

n	cond (H_n)	n	cond (H_n)
3	748	12	$4.1154 \cdot 10^{16}$
4	28375	13	$1.3244 \cdot 10^{18}$
5	$9.4366 \cdot 10^{5}$	14	$4.5378 \cdot 10^{19}$
6	$2.9070 \cdot 10^{7}$	15	$1.5392 \cdot 10^{21}$
7	$9.8519 \cdot 10^{8}$	16	$5.0628 \cdot 10^{22}$
8	$3.3873 \cdot 10^{10}$	17	$1.6808 \cdot 10^{24}$
9	$1.0996 \cdot 10^{12}$	18	$5.7661 \cdot 10^{25}$
10	$3.5357 \cdot 10^{13}$	19	$1.9258 \cdot 10^{27}$
11	$1.2337 \cdot 10^{15}$	20	$6.2836 \cdot 10^{28}$

Таблица 2 – Решения СЛАУ с матрицами H_n.

n=10	n=12	n=14	n=20
1.000000	1.000000	1.000000	1.000000
1.000000	1.000003	0.999990	1.000075
0.999998	0.999916	1.000310	0.997232
1.000020	1.001125	0.996220	1.043007
0.999903	0.991859	1.019948	0.657461
1.000267	1.035385	0.981738	2.485524
0.999563	0.902315	0.675197	-2.218021
1.000421	1.175419	2.922896	2.012694
0.999779	0.795768	-4.498100	11.914026
1.0000480	1.148663	10.515965	-20.728838
	0.938525	-9.428208	5.732215
	1.011022	8.094763	34.233277
		-1.740964	-42.153623
		1.460245	18.727738
			2.721547
			-16.803379
			49.405743
			-57.516277
			33.288344
			-5.798746

3. ЧИСЛЕННОЕ РЕШЕНИЕ ПЛОХО ОБУСЛОВЛЕННОЙ СИСТЕМЫ УРАВНЕНИЙ

Различные подходы к решению систем уравнений с плохо обусловленными матрицами известны (см. [3-9]). В данной работе для получения приемлемого решения СЛАУ рассматривается применение модификации метода регуляризации. Известный стандартный метод регуляризации Тихонова позволяет найти нормальное решение системы $Az = f$, т.е. такой элемент z, который минимизирует величину $\| Az - f \|$ и имеет наименьшую норму; он существует и определяется однозначно. Этот

метод основан на поиске элемента, на котором функционал $M_\alpha(z, A, f) = \|Az - f\|^2 + \alpha \|z\|^2$ достигает наименьшего значения для фиксированного положительного α. Для этого необходимо решить уравнение Эйлера $(A^*A + \alpha E)z = A^*f$, где A^* - сопряженная матрица. Нормальное решение находится как предел таких элементов при стремлении параметра α к нулю по определенному закону, определяемому точностью задания исходной матрицы A и правой части f. Решение уравнения Эйлера зависит от числа обусловленности матрицы A^*A. Это число может быть очень большим. Если матрица A симметрична и положительно определена, мы предлагаем найти нормальное решение другим способом. В данной работе предлагается найти нормальное решение путем решения системы уравнений, для которой число обусловленности значительно меньше.

Пусть матрица СЛАУ

$$Az = f \tag{1}$$

является симметричной и положительно определенной (например, матрица Гильберта). В этом случае существует единственный положительно определенный корень из матрицы $B = \sqrt{A}$, т.е. положительно определенная матрица B такая, что $B^2 = A$. Установим связь между собственными значениями и собственными векторами матриц A и B: пусть μ и $x -$ собственное значение и собственный вектор матрицы B:

$$Bx = \mu x. \tag{2}$$

Умножив (2) на B, получаем

$$Ax = \mu Bx \tag{3}$$

Используя (2), перепишем (3) как $Ax = \mu^2 x$. Это равенство означает, что собственные векторы матриц A и B одинаковы, а собственные значения

матрицы A равны квадратам собственных значений матрицы B. Умножив (1) на B^{-1} , получаем

$$Bz=B^{-1}f. \tag{4}$$

Запишем для (4) уравнение Эйлера для минимизации сглаживающего функционала $M_\alpha(z,B,B^{-1}f)=\|Bz-B^{-1}f\|^2+\alpha\|z\|^2$: оно имеет вид

$$(B^*B+\alpha E)z_\alpha=B^*(B^{-1}f), \qquad \alpha>0. \tag{5}$$

В случае симметричной матрицы A матрица B является самосопряженной, поэтому мы получаем уравнение

$$(A+\alpha E)z_\alpha=f. \tag{6}$$

Формально в исходной системе осуществляется сдвиг, но фактически это метод регуляризации для уравнения (1).

В памяти компьютера числа обычно хранятся с некоторыми погрешностями. Будем считать, что матрица и вектор даны приближенно, т.е. вместо матрицы A и правой части f известны A_δ и f_δ такие, что $\|A-A_\delta\|\le\delta$, $\|f-f_\delta\|\le\delta$. Говорить о сходимости можно лишь при неограниченном увеличении точности исходной информации, т.е. при δ→0. Однако практически мы не можем неограниченно уменьшать δ, из-за чего результаты могут быть далеки от желаемых решений. Решая задачу (6), получаем приближенное решение системы (1).

Замечание. Методы вычисления квадратного корня матрицы описаны в [11-13]. Если матрица A симметрична, нам не нужно вычислять матрицу B. Применение регуляризации непосредственно к уравнению (1) просто увеличит число обусловленности результирующей системы, что невыгодно. В случае несимметричной положительно определенной матрицы уравнение Эйлера для системы (1) имеет вид (5). Сходимость метода регуляризации изложена в [3]. В отличие от стандартного подхода процедуры

регуляризации представление (5) имеет меньшее число обусловленности, что очень важно. Но, в отличие от симметричных матриц, необходимо знать матрицу B в случае несимметричной матрицы. Последнее требование усложняет применение данного метода на практике.

Существуют классы несимметричных матриц, все собственные значения которых положительны. Покажем, как в этом случае можно осуществить процесс регуляризации в форме (5), а не в традиционной форме $(A^*A+\alpha E)z=A^*f,\ \alpha>0,$ что, как уже упоминалось выше, существенно уменьшает число обусловленности решаемой системы. Рассматривается класс осцилляционных матриц, появляющихся при исследовании малых колебаний механических систем, все собственные значения которых положительны и попарно различны, а соответствующие собственные векторы обладают особыми свойствами колебаний (см. [14]).

Пусть матрица A_n- осцилляционная, а V есть матрица собственных векторов $A_n,\ \Lambda=[\lambda_1,\dots,\lambda_n]$ – диагональная матрица собственных чисел A_n. Тогда $A_nV=VA_n$ или $A_n=V\,\Lambda V^{-1}$. Пусть $\Lambda_1=[\sqrt{\lambda_1},\dots,\sqrt{\lambda_n}]$. Положим $B=V\Lambda_1V^{-1}$. Понятно, что $B^2=V\Lambda_1V^{-1}V\Lambda_1V^{-1}=V\Lambda_1^2V^{-1}=V\Lambda V^{-1}=A_n$. Таким образом, процесс регуляризации может быть осуществлен в форме (5). Обобщенная матрица Вандермонда, то есть матрица вида $\left(a_i^{b_k}\right)_{i,k=1}^{n}$, $0<a_1<\dots<a_n$, $b_1<\dots<b_n$, относится к классу осцилляционных матриц. Для регуляризации СЛАУ с такими матрицами применима описанная выше схема.

4. РЕЗУЛЬТАТЫ ПРИМЕНЕНИЯ МЕТОДА РЕГУЛЯРИЗАЦИИ

Проведена серия вычислительных экспериментов по применению метода регуляризации для решения СЛАУ с матрицами Гильберта порядков n=2,3,...,20. В таблицаз 3, 4 показаны результаты применения метода регуляризации для различных параметров $\alpha=10^{-1},10^{-2},\dots,10^{-12}$ для

решения возмущенной СЛАУ $(H_n + \alpha E_n)z = f$, где матрица исходной системы $H_n = (h_{ij})_{i,j=1}^{n}$ есть матрица Гильберта порядка n. Решение возмущенной системы получено методом Гаусса с выбором ведущего элемента по столбцам. Точное решение исходной невозмущенной системы $H_n z = f$ есть n-мерный вектор из единиц: $z = (1.0,\ 1.0,\ \ldots, 1.0)^T$. Вычисляя решение возмущенной СЛАУ при различных значениях α, находим значение параметра, при котором погрешность решения имеет минимальное значение. Параметр α, соответствующий решению с наименьшей нормой, назовем оптимальным. Таким образом, для решения системы уравнений (1) необходимо решить несколько систем уравнений для различных α. В таблицах 3, 4 полужирным шрифтом выделено наименьшее значение нормы погрешности для заданного n, соответствующее оптимальному значению параметра возмущения α. В последней строке этих таблиц приведены нормы погрешностей решений, полученных без регуляризации. Погрешность вычислялась в евклидовой норме.

Таблица 3 – Нормы погрешностей решений СЛАУ с матрицами Гильберта порядка $n = 10, 12$ методом (6) при различных значениях параметра α.

α	$n = 10$	$n = 12$
10^{-12}	0.1310^{-5}	$0.22 \cdot 10^{-5}$
$\mathbf{10^{-11}}$	$\mathbf{0.14 \cdot 10^{-6}}$	$\mathbf{0.69 \cdot 10^{-7}}$
10^{-10}	$0.16 \cdot 10^{-6}$	$0.19 \cdot 10^{-6}$
10^{-8}	$0.16 \cdot 10^{-3}$	$0.20 \cdot 10^{-3}$
10^{-7}	$0.58 \cdot 10^{-3}$	$0.58 \cdot 10^{-3}$
10^{-6}	$0.17 \cdot 10^{-2}$	$0.19 \cdot 10^{-2}$
10^{-5}	$0.56 \cdot 10^{-2}$	$0.63 \cdot 10^{-2}$
10^{-4}	$0.18 \cdot 10^{-1}$	$0.19 \cdot 10^{-1}$
10^{-3}	$0.56 \cdot 10^{-1}$	$0.61 \cdot 10^{-1}$
10^{-2}	0.1799	0.1985
10^{-1}	0.5788	0.6355
Нормы погрешностей решений без применения метода регуляризации		
	$0.71 \cdot 10^{-3}$	0.3306

Точность арифметики с плавающей запятой можно охарактеризовать машинным ε, т.е. наименьшим положительным числом с плавающей запятой ε, для которого 1 + ε >1 (см. [8]). Мы можем вычислить машинное ε. Например, в 64-битном C++ переменные типа double дают ε $\approx 2.2 \cdot 10^{-16}$.

Таблица 4 – Нормы погрешностей решения СЛАУ с матрицами Гильберта порядков $n = 14, 20$ методом (6) при различных значениях параметра α.

α	$n = 14$	$n = 20$
10^{-12}	$0.19 \cdot 10^{-4}$	$0.17 \cdot 10^{-4}$
10^{-11}	$0.27 \cdot 10^{-6}$	$\mathbf{0.23 \cdot 10^{-6}}$
10^{-10}	$\mathbf{0.20 \cdot 10^{-6}}$	$0.25 \cdot 10^{-6}$
10^{-8}	$0.20 \cdot 10^{-3}$	$0.25 \cdot 10^{-3}$
10^{-7}	$0.65 \cdot 10^{-3}$	$0.78 \cdot 10^{-3}$
10^{-6}	$0.21 \cdot 10^{-2}$	$0.25 \cdot 10^{-2}$
10^{-5}	$0.66 \cdot 10^{-2}$	$0.80 \cdot 10^{-2}$
10^{-4}	$0.21 \cdot 10^{-1}$	$0.25 \cdot 10^{-1}$
10^{-3}	$0.67 \cdot 10^{-1}$	$0.81 \cdot 10^{-1}$
10^{-2}	0.2153	0.2581
10^{-1}	0.6872	0.8220
Погрешности решении без регуляризации		
	17.0703	105.2819

Теорема Тихонова утверждает, что теоретически, по мере уменьшения α, регуляризованное решение улучшается, но в практических расчетах для достаточно малых α (в пределах точности машины в C++) погрешности округления и число обусловленности матрицы имеют значительное влияние. Это можно увидеть, изучив результаты, представленные в начале таблиц 3, 4.

Заключение

В данной работе представлены результаты численного решения СЛАУ с положительно определенными симметричными (или несимметричными, но почти симметричными) плохо обусловленными матрицами модифицированным методом регуляризации. Показано, что решение СЛАУ с матрицами Гильберта с использованием метода регуляризации может быть существенно улучшено [15].

Задачи и упражнения

1. Разработать программу на MAPLE для решения СЛАУ с матрицей Гильберта.
2. Включить в программу блок с нахождением собственных чисел матрицы, а также числа обусловленности матрицы.
3. Определить, сколько нужно брать цифр в мантиссе (Digits), чтобы собственные числа матрицы определялись правильно при выбранном размере матрицы.
4. Решить СЛАУ методом регуляризации и обычным методом. Сравнить результаты.

Список литературы

1. Даугавет И.К. Введение в классическую теорию приближения функций. – СПб.: Изд-во СПбГУ, 2011. – 232 с.
2. Михлин С.Г. Численная реализация вариационных методов. М.: Наука. Главная редакция физико-математической литературы, 1966. – 432 с.
3. Тихонов А. Н. Методы решения некорректных задач / А. Н. Тихонов, В. Я. Арсенин. – М.: Наука. Главная редакция физико-математической литературы, 1979. Изд. 2-е. – 284 с.
4. Фаддеев Д. К. Вычислительные методы линейной алгебры / Д. К. Фаддеев, В. Н. Фаддеева. – Электронно-библиотечная система Издательства Лань, 2009. – 753 с.
5. Фаддеев Д.К. О плохо-обусловленных системах линейных уравнений / Д. К. Фаддеев, В. Н. Фаддеева // Журнал вычислительной математики и математической физики. – 1961. – Т. 1. – №3. – С. 412–417.
6. Гавурин М.К. О плохо-обусловленных системах линейных алгебраических уравнений / М. К. Гавурин // Журнал вычислительной математики и математической физики. – 1962. – Т. 2. – №3. – С. 387–397.
7. Гавурин М. К. Применение полиномов Чебышева при регуляризации некорректных и плохо обусловленных уравнений в гильбертовом пространстве / М. К. Гавурин, В. М. Рябов // Журнал вычислительной математики и математической физики. – 1973. – Т. 13. – №6. – С. 1599–1601.
8. Форсайт Дж. Численное решение систем линейных алгебраических уравнений / Дж. Форсайт, К. Молер. Перевод

с английского В.П. Ильина и Ю.И. Кузнецова. Под редакцией Г.И. Марчука. – М.: Мир, 1969. – 167 с.

9. Кабанихин С. И. Обратные и некорректные задачи / С.И. Кабанихин. – Новосибирск: Сибирское научное издательство, 2009. – 457 с.
10. Bjorck A. A Schur method for the square root of a matrix / A. Bjorck, S. Hammarling // Linear Algebra Appl. – 1983. – V. 52/53, P. 127–140.
11. Higham Nicholas J. Functions of matrices: Theory and computation / J. Higham Nicholas – Philadelphia: Society for Industrial and Applied Mathematics, 2008. – 425 p.
12. Deadman E. Blocked Schur algorithms for computing the matrix square root / E. Deadman, N.J. Higham, R. Ralha // Lecture Notes in Computer Science, V. 7782, Springer-Verlag. – 2012. P. 171–182.
13. Higham Nicholas J. Newton's Method for the Matrix Square Root / J. Higham Nicholas // Mathematics of computation. – 1986. – V. 46. – №174. – P. 537–549.
14. Гантмахер Ф. Р. Осцилляционные матрицы и ядра и малые колебания механических систем / Ф.Р. Гантмахер, М.Г. Крейн // Москва-Ленинград: Государственное издательство технико-теоретической литературы [ГИТТЛ], 1950. 359 с.
15. Бурова, И. Г., Рябов, В. М., Кальницкая, М. А., Малевич, А. В. & Демьянович, Ю. К. Программа для решения системы линейных алгебраических уравнений с положительно определенной матрицей методом регуляризации // Патент № 2018661356 , 6 сен 2018.

Приложение 1

Программа решения системы линейных уравнений методом Гаусса без выбора ведущего элемента (Maple)

Данная программа предназначена для решения системы линейных алгебраических уравнений (СЛАУ) методом Гаусса по схеме единственного деления.
N - размер квадратной матрицы СЛАУ,
A - матрица системы линейных алгебраических уравнений,
B - вектор правой части СЛАУ,
X - вектор, в котором будет записано решение системы.

```
> restart; Digits:=15; with(LinearAlgebra):
> N:=3;
OUT_DATA:=fopen("F:\\TEMP_REZ\\N3_bv_maple_d15.txt", WRITE):
#файл для записи результата

A:=Matrix(N,N):
# Ввод матрицы A из файла (необходимо указать полное имя файла):
MAF:=readdata("F:\\GBM\\GbM_3.txt", N); A:=convert(MAF, Matrix);

X:=Vector(N):
B:=Vector(N):
# Ввод вектора свободных членов из файла (необходимо указать полное имя
файла):
VBF:=readdata("F:\\GBM\\B_25.txt", float, N); #путь к файлу с вектором B
B:=convert(VBF, Vector):

> for k to N
do print('Step ', k);
Ved[k]:=A[k,k];
#Сообщение об ошибке
if Ved[k]=0 then print ("ВЕДУЩИЙ ЭЛЕМЕНТ РАВЕН НУЛЮ !!!!");
`quit`(12); fi;
for j to N do A[k,j]:=A[k,j]/Ved[k]; od;
B[k]:=B[k]/Ved[k];
for i from k+1 to N
do MIK:=A[i,k];
for j to N do A[i,j]:=A[i,j]-A[k,j]*MIK; od;
B[i]:=B[i]-B[k]*MIK;
od;
```

```
od;

>
#ОБРАТНЫЙ ХОД МЕТОДА ГАУССА
for k from N by -1 to 1
do X[k]:=B[k]:
for j from N by -1 to k+1 do X[k]:=X[k]-A[k,j]*X[j]: od;
od;
print("Решение системы AX=B : "); X;

#ЗАПИСЬ РЕШЕНИЯ В ФАЙЛ:
#файл открыт в начале программы
Z:=convert(X,list);
writedata(OUT_DATA, Z);
fclose(OUT_DATA);
```

Приложение 2

Программа решения системы линейных уравнений методом Гаусса с выбором ведущего элемента по столбцу (Maple)

Данная программа предназначена для решения системы линейных уравнений методом Гаусса с выбором ведущего элемента по столбцу.
N - размер квадратной матрицы,
A - матрица системы уравнений,
B - правая часть системы уравнений,
X - вектор, в котором будет записано решение системы,
Ved - вектор, состоящий из ведущих элементов, выбранных на k-м шаге решения,
PS - вектор, k-я компонента которого является номером строки ведущего элемента на k-м шаге.

```
> restart; Digits:=15; with(LinearAlgebra):
> N:=3;
A:=Matrix(N,N): A1:=Matrix(N,N):#A1-копия матрицы A
X:=Vector(N):
B:=Vector(N): B1:=Vector(N): #B1-копия вектора B
Ved:=Vector(N):
PS:=Vector(N): for i to N do PS[i]:=i: od: PS;

OUT_DATA:=fopen("F:\\TEMP_REZ\\N3_col_maple_d15.txt", WRITE):
#файл для записи результата

#Ввод матрицы A из файла (необходимо указать полное имя файла):
>MAF:=readdata("F:\\GBM\\GbM_3.txt", N); A:=convert(MAF, Matrix);

#Ввод вектора свободных членов из файла (необходимо указать полное имя файла):
> VBF:=readdata("F:\\GBM\\B_25.txt", float, N);
B:=convert(VBF, Vector):

> for k to N
do print('Step ', k);
Ved[k]:=A[k,k];
for i from k+1 to N # поиск ведущего элемента
do if abs(A[i,k])>abs(Ved[k])
then Ved[k]:=A[i,k]; PS[k]:=i;
fi;
od;
```

```
#Сообщение об ошибке
if Ved[k]=0 then print ("ВЕДУЩИЙ ЭЛЕМЕНТ РАВЕН НУЛЮ !!!!");
`quit`(12); fi;

#перестановка строк матрицы:
for j to N do Elem:=A[PS[k],j]; A[PS[k],j]:=A[k,j]; A[k,j]:=Elem; od;
Elem:=B[PS[k]]; B[PS[k]]:=B[k]; B[k]:=Elem;
#перестановка соответствующих элементов вектора свободных членов
for j to N do A[k,j]:=A[k,j]/Ved[k]; od;
B[k]:=B[k]/Ved[k];
for i from k+1 to N
do MIK:=A[i,k];
for j to N do A[i,j]:=A[i,j]-A[k,j]*MIK; od;
B[i]:=B[i]-B[k]*MIK;
od;
od;

#ОБРАТНЫЙ ХОД МЕТОДА ГАУССА

for k from N by -1 to 1
do X[k]:=B[k];
for j from N by -1 to k+1 do X[k]:=X[k]-A[k,j]*X[j]; od;
od;
print("Решение системы AX=B : "); X;

#ЗАПИСЬ РЕШЕНИЯ В ФАЙЛ:
#файл для записи результатов открыт в начале программы

Z:=convert(X,list);
writedata(OUT_DATA, Z);
fclose(OUT_DATA);
```

Приложение 3

Программа решения системы линейных уравнений методом Гаусса без выбора ведущего элемента (С)

Решение системы линейных уравнений методом Гаусса (схема единственного деления)
A - матрица системы, B – вектор свободных членов, X – вектор решения.
Матрица A считывается из файла.

```
#include "stdafx.h"
#include <stdio.h>
#include <math.h>
#include <iostream>
#include <string>
#include <conio.h>
using namespace std;

int main()
{
#define n 13

double A[n][n];
double B[n],X[n];
double mik[n], Sum, s[n];

int k, i, j, q;
//string Path="G:\\GBM\\GbM_10.txt";

FILE *file;
fopen_s(&file, "G:\\GBM\\GbM_13.txt", "r");
//путь к файлу с матрицей Гильберта

for (i = 0; i < n; i++)
{
for (j = 0; j < n; j++)
{
if (fscanf_s(file, "%lf", &(A[i][j])) != EOF)
{
printf("%lf \n", A[i][j]);
}
}
}
```

```
fclose(file);

//Вычисление правой части системы AX=Ae=B
for (i=0; i<n; i++) {B[i] = 0.00;}
for (i=0; i<n; i++)
for (j=0; j<n; j++) {B[i] = B[i] + A[i][j] ;}
for (i=0; i<n; i++) {cout <<"B["<<i<<"]="<<B[i]<<'\n';}

//Прямой ход метода Гаусса
for (k = 1; k <= n - 1; k++)
{
if (A[k - 1][k - 1] != 0)
for (i = k + 1; i <= n; i++)
mik[i - 1] = A[i - 1][k - 1] / A[k - 1][k - 1];
else { printf("Ошибка в данных: деление на ноль \n"); goto FFF; };

for (i = k + 1; i <= n; i++)
{
for (j = k; j <= n; j++)
{A[i - 1][j - 1] = A[i - 1][j - 1] - A[k - 1][j - 1] * mik[i - 1];
}
B[i-1]=B[i-1]-mik[i-1]*B[k-1];
}
}

//Обратный ход метода Гаусса
for (i=0; i<n; i++) X[i]=0.00;
X[n-1]=B[n-1]/A[n-1][n-1];
cout<<X[n-1]<<'\n';
for (k=n-2;k>=0;k--)
{Sum=0.00;
for (j=k+1; j<n; j++) Sum+=X[j]*A[k][j];
X[k]=(B[k]-Sum)/A[k][k];
}
//Вывод вектора X на экран
for (i=0;i<n;i++) cout<<"X["<<i<<"]= "<<X[i]<<'\n';
//

FFF: return 0;
}
```

Приложение 4

Программа решения системы линейных уравнений методом Гаусса с выбором ведущего элемента по всей матрице (С)

Решение системы линейных алгебраических уравнений методом Гаусса с выбором ведущего элемента по всей матрице, с соответствующими перестановками строк и столбцов и перестановкой элементов вектора решения.
MA - расширенная матрица системы, считывается из файла.
N – размер матрицы,
PL[N], PI[N] – векторы, в которых записаны номера строк и столбцов ведущих элементов на каждой итерации;
V[N] – коэффициенты, на которые умножаются элементы ведущей строки при вычитании ее из строк матрицы для обнуления поддиагональных элементов;
X[N] – вектор решения.

```
#include<stdio.h>
#include <time.h>
#include <math.h>

//using namespace std;

//#include <mpi.h>

#define N 5

double MA[N][N + 1], V[N], X[N];
int PL[N], PI[N];

int main(int args, char **argv)
{
   FILE *INF_MA;
   INF_MA= fopen ("/home/marina/F_19_FEB/ma_5_12345.txt", "r");


   // Считываем матрицу MA из файла
   printf ("\n MATRIX MA: ");
   for (int i = 0; i < N; i++)
   {  printf ("\n");
      for (int j = 0; j <= N; j++)
      {
```

```
        if (fscanf (INF_MA, " %lf", &MA[i][j] ) !=EOF)
            printf(" MA[%d][%d]= %f", i, j, MA[i][j]);

      };
      printf ("\n");
  };

fclose(INF_MA);

  //вывод считанной матрицы
 printf("\n ВЫВОД введенной из файла ma_%d.txt МАТРИЦЫ: \n",  N);
for (int i=0;i<N; i++)
        {for (int j=0;j<=N; j++)
                {
                        printf (" %f  ", MA[i][j]);
                };
         printf("\n");
        };

  //гипотетические начальные значения строк и столбцов ведущих элементов
  for (int i = 0; i<N; i++) {PL[i]=i; PI[i]=i; };

// Прямой ход
for (int iteration = 0; iteration < N - 1; iteration++)
{
  // ищем главный элемент по всей необработанной части матрицы
  int maxLine=iteration;
  int maxIndex=iteration;
  double max=MA[iteration][iteration];

  for (int i = iteration; i < N; i++)
  {
    for (int j = iteration; j < N; j++)
    {
      if (fabs(MA[i][j]) > max)
      {
        max = MA[i][j];  maxLine = i;   maxIndex = j;
      }
    }
  };

  PL[iteration]=maxLine;
  PI[iteration]=maxIndex;
```

```
printf ("НА итерации %d ведущий элемент MA[ %d ] [ %d ] = %f \n", iteration, maxLine, maxIndex, max);

    //меняем местами с верхней строкой
    if (PL[iteration] != iteration)
    {
       for (int j = 0; j <= N; j++)  //было < а теперь <=  !!!
       {
         double buff = MA[iteration][j];
         MA[iteration][j] = MA[maxLine][j];
         MA[maxLine][j] = buff;
       };

     };// конец перемены строк

//меняем местами столбцы
   if (PI[iteration] != iteration )
   { for (int i = 0; i < N; i++)
       {
          double buff = MA[i][iteration];
          MA[i][iteration] = MA[i][maxIndex];
          MA[i][maxIndex] = buff;
       };

   }// конец перестановки столбцов


    //вычисляем множители
    for (int i = iteration + 1; i < N; i++)
    {
      V[i] = MA[i][iteration] / MA[iteration][iteration];
    };

    //вычитаем главную строку
    for (int i = iteration + 1; i < N; i++)
    {
      for (int j = 0; j <= N; j++)
      {
        MA[i][j] = MA[i][j] - V[i] * MA[iteration][j];
      }
    };

  } //конец прямого хода
```

```
   //конец прямого хода //вывод  матрицы
   printf("\n ПРЯМОЙ ХОД ЗАВЕРШЕН ВЫВОД МАТРИЦЫ: \n");
  for (int i=0;i<N; i++)
         {for (int j=0;j<=N; j++)
                {
                        printf (" %f  ", MA[i][j]);
                };
          printf("\n");
         }

//обратный ход
    for (int k = N-1; k>=0; k--)
        {
           X[k]=MA[k][N];
           for (int j= N-1; j>=k+1; j--)
              {
                 X[k]-=MA[k][j];
              };
           X[k]=X[k]/MA[k][k];
       };

 for (int iter=N-1; iter>=0; iter--) //перестановка элементов вектора решения
    {
     double buf=X[iter]; X[iter]=X[PI[iter]]; X[PI[iter]]=buf;
      } ;

   printf("\nResult NEW: \n");
  for (int i = 0; i < N; i++) { printf(" %f \n", X[i]); }

 return(0);
}
```

Приложение 5

Программа для решения системы линейных алгебраических уравнений с положительно определенной симметричной матрицей методом регуляризации (SLAE-RjaBKM)
(Свидетельство о государственной регистрации программы для ЭВМ № 2018661356)

```
#include <stdio.h>
#include <math.h>

void about(void)
{
printf("************************************************************\n");
printf("| ");
printf("------------\"SLAE_Ryabov_RP\"-------------\n");
printf("| ");
printf(" (SLAE Ryabov Regularization Process) \n");
printf("| ");
printf("Авторы: Рябов В.М., Бурова И.Г., Кальницкая М.А., Малевич А.В.
\n");
printf("| ");
printf("Санкт-Петербургский государственный университет, 2018 (c) \n");
printf("| ");
printf("Описание: Программа для решения плохо обусловленных систем
линейных \n");
printf("| ");
printf("алгебраических уравнений (СЛАУ) с положительно определенными
\n");
printf("| ");
printf("матрицами методом регуляризации Рябова В.М. \n");
printf("************************************************************\n");
}

int gauss_method(const int n, double A_exp[n][n+1], double col_res[n])
{
int k, i, j;
double first, sum, s[n];
// Прямой ход метода Гаусса
for (k = 0; k < n-1; k++)
{
s[k] = A_exp[k][k];

for (i = k + 1; i < n; i++)
```

```
{
first = A_exp[i][k] / s[k];
for (j = k; j <= n; j++)
{
A_exp[i][j] -= A_exp[k][j] * first;
}
}
}

// Обратный ход метода Гаусса
for (i = n - 1; i >= 0; i--)
{
sum = 0.00;

for (j = i + 1; j < n; j++)
{
sum += A_exp[i][j] * col_res[j];
}
col_res[i] = (A_exp[i][n] - sum) / A_exp[i][i];
}
return 0;
}

int main ()
{
about();
int n;
printf("1. Введите размер системы, n = ");
scanf("%d", &n);

double A[n][n], A_exp_1[n][n+1], A_exp_2[n][n+1], col_free_start[n],
col_free_end[n], col_free_end_2[n], col_res[n], col_res_2[n], s[n];
double sum, first, NORM, NORM_2, alpha;
int si[n], sj[n];
int i, j, k, m, z, x, y, w;
char A_file[64], col_free_file[64];
printf("2. Введите параметр регуляризации, альфа = ");
scanf("%lf", &alpha);
getchar();
printf("3. Файл с матрицей системы: ");
gets(A_file);
//
// Читаем матрицу из файла
```

```
//
FILE *file;
file = fopen(A_file, "r");
// Проверка открытия файла
if (file == NULL) printf ("Ошибка открытия файла! \n");
else
{
printf ("Файл успешно открыт! Enter - продолжить ... \n");

for (i = 0; i < n; i++)
{
for (j = 0; j < n; j++)
{
if (fscanf(file, "%lf", &(A[i][j])) != EOF) {}
}
}
}

fclose(file);
getchar();
printf("4. Файл с вектором свободных членов: ");
gets(col_free_file);
//
// Читаем вектор из файла
//
FILE *file_2;
file_2 = fopen(col_free_file, "r");
// Проверка открытия файла
if (file == NULL) printf ("Ошибка открытия файла! \n");
else
{
printf ("Файл успешно открыт! Enter - продолжить ... \n");

for (i = 0; i < n; i++)
{
if (fscanf(file_2, "%lf", &(col_free_start[i])) != EOF) {}
}
}
fclose(file_2);
getchar();

//
// Генерируем расширенную матрицу без возмущения
```

```
//
for (i = 0; i < n; i++)
{
for (j = 0; j < n; j++)
{
A_exp_1[i][j] = A[i][j];
}

A_exp_1[i][n] = col_free_start[i];
}

// Решение СЛАУ методом Гаусса
gauss_method(n, A_exp_1, col_res);
//
// Вычисляем столбец свободных членов после решения
//
for (i = 0; i < n; i++)
{
sum = 0.00;
col_free_end[i] = 0.00;
for (j = 0; j < n; j++)
{
sum += A[i][j]*col_res[j];
}
col_free_end[i] = sum;
}
//
// Вычисляем эвклидову норму невязки без возмущения
//
NORM=0.0;
for (i = 0; i < n; i++)
{
NORM+=(col_free_end[i]-col_free_start[i])*(col_free_end[i]-col_free_start[i]);
}
NORM = sqrt(NORM);
//
// Генерируем расширенную матрицу с возмущением
//
for (i = 0; i < n; i++)
{
for (j = 0; j < n; j++)
{
A_exp_2[i][j] = A[i][j];
```

```
}

A_exp_2[i][n] = col_free_start[i];
}
//
// Выполняем возмущение матрицы
//
for (i = 0; i < n; i++)
{
A_exp_2[i][i] += alpha;
}
// Решение СЛАУ методом Гаусса
gauss_method(n, A_exp_2, col_res_2);
//
// Вычисляем столбец свободных членов после решения
//
for (i = 0; i < n; i++)
{
sum = 0.00;
col_free_end_2[i] = 0.00;
for (j = 0; j < n; j++)
{
sum += A[i][j]*col_res_2[j];
}
col_free_end_2[i] = sum;
}
//
// Вычисляем эвклидову норму невязки при возмущении
//
NORM_2=0.0;

for (i = 0; i < n; i++)
{
NORM_2+=(col_free_end_2[i]-col_free_start[i])*(col_free_end_2[i]-
col_free_start[i]);
}

NORM_2 = sqrt(NORM_2);
//
// Выводим норму невязки на экран
//
printf("Вектор решения до и после регуляризации: \n");
for (i = 0; i < n; i++)
```

```
{
printf("x[%d]=%20.8e ", i, col_res[i]);
printf("x[%d]=%20.8e \n", i, col_res_2[i]);
}
printf("Норма_невязки=%.4E ", NORM);
printf(" Норма_невязки=%.4E \n\n", NORM_2);
return 0;
}
```

Результат работы программы для решения системы линейных алгебраических уравнений с положительно определенной симметричной матрицей методом регуляризации (SLAE-RjaBKM)

```
*************************************************************************
| ------------"SLAE_Ryabov_RP"--------------
|      (SLAE Ryabov Regularization Process)
| Авторы: Рябов В.М., Бурова И.Г., Кальницкая М.А., Малевич А.В.
| Санкт-Петербургский государственный университет, 2018 (c)
| Описание: Программа для решения плохо обусловленных систем линейных
| алгебраических уравнений (СЛАУ) с положительно определенными
| матрицами методом регуляризации Рябова В.М.
*************************************************************************
1. Введите размер системы, n = 20
2. Введите параметр регуляризации, альфа = 1e-15
3. Файл с матрицей системы: hilbert_20.txt
Файл успешно открыт! Enter - продолжить ...

4. Файл с вектором свободных членов: b_20.txt
Файл успешно открыт! Enter - продолжить ...

Вектор решения до и после регуляризации:
x[0]=      9.99999502e-01  x[0]=      9.99999992e-01
x[1]=      1.00007510e+00  x[1]=      1.00000092e+00
x[2]=      9.97232353e-01  x[2]=      9.99974315e-01
x[3]=      1.04300725e+00  x[3]=      1.00028506e+00
x[4]=      6.57461072e-01  x[4]=      9.98622401e-01
x[5]=      2.48552379e+00  x[5]=      1.00153089e+00
x[6]=     -2.21802057e+00  x[6]=      1.01280772e+00
x[7]=      2.01269374e+00  x[7]=      9.40307521e-01
x[8]=      1.19140261e+01  x[8]=      1.10099198e+00
x[9]=     -2.07288375e+01  x[9]=      9.66389639e-01
x[10]=      5.73221531e+00  x[10]=      8.77968303e-01
x[11]=      3.42332773e+01  x[11]=      1.16327546e+00
x[12]=     -4.21536232e+01  x[12]=      9.41086502e-01
x[13]=      1.87277379e+01  x[13]=      1.00517849e+00
x[14]=      2.72154695e+00  x[14]=      9.95643269e-01
x[15]=     -1.68033786e+01  x[15]=      9.70280504e-01
x[16]=      4.94057426e+01  x[16]=      9.92349159e-01
x[17]=     -5.75162772e+01  x[17]=      1.09894441e+00
x[18]=      3.32883435e+01  x[18]=      9.09220785e-01
x[19]=     -5.79874570e+00  x[19]=      1.02514269e+00
Норма_невязки=1.2363E-15  Норма_невязки=3.7072E-15
```

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2018661356

«Программа для решения системы линейных алгебраических уравнений с положительно определенной симметричной матрицей методом регуляризации» (SLAE-RjaBKM)

Правообладатель: ***федеральное государственное бюджетное образовательное учреждение высшего образования "Санкт-Петербургский государственный университет" (СПбГУ) (RU)***

Авторы: ***см. на обороте***

Заявка № **2018618523**
Дата поступления **09 августа 2018 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***06 сентября 2018 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

Авторы: ***Рябов Виктор Михайлович (RU), Бурова Ирина Герасимовна (RU), Кальницкая Марина Алексеевна (RU), Малевич Александр Владиславович (RU), Демьянович Юрий Казимирович (RU)***